Advances in Difference Equations
a SpringerOpen Journal

RESEARCH Open Access

A compact finite difference method for reaction–diffusion problems using compact integration factor methods in high spatial dimensions

Rongpei Zhang[1], Zheng Wang[2*], Jia Liu[3] and Luoman Liu[1]

*Correspondence:
wangzhen_sd@126.com
[2]College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao, P.R. China
Full list of author information is available at the end of the article

Abstract

This paper proposes and analyzes an efficient compact finite difference scheme for reaction–diffusion equation in high spatial dimensions. The scheme is based on a compact finite difference method (cFDM) for the spatial discretization. We prove that the proposed method is asymptotically stable for the linear case. By introducing the differentiation matrices, the semi-discrete reaction–diffusion equation can be rewritten as a system of nonlinear ordinary differential equations (ODEs) in matrices formulations. For the time discretization, we apply the compact implicit integration factor (cIIF) method which demands much less computational effort. This method combines the advantages of cFDM and cIIF methods to improve the accuracy without increasing the computational cost and reducing the stability range. Numerical examples are shown to demonstrate the accuracy, efficiency, and robustness of the method.

Keywords: Compact implicit integration factor methods; Compact finite difference method; Stiff reaction–diffusion equations; High spatial dimensions

1 Introduction

This paper is concerned with high-dimensional reaction–diffusion systems of the following form:

$$u_t = \mathcal{D}\Delta u + \mathcal{F}(u), \tag{1}$$

where $u \in \mathbf{R}^m$ represents concentration of m types of molecules or chemical species, $\mathcal{D}$ is the matrix of diffusion coefficients, and $\mathcal{F}(u)$ represents reactions and interactions among different species. The boundary condition is considered to be periodic boundary condition. It can also apply to other boundary conditions.

Well-known examples of reaction–diffusion systems include the Schnakenberg model [14], the chloride-iodide-malonic acid (CIMA) reactive model [8], the Gray–Scott model [4], the Gierer–Meinhardt model [3]. Efficient and accurate simulation of such systems (1), however, is difficult. This is because they couple a stiff diffusion term with a (typically) strongly nonlinear reaction term. When discretized, this leads to large systems of strongly

nonlinear, stiff ODEs. A class of efficient implicit integration factor (IIF) methods [12] was developed for implicit treatment of the stiff reactions. In the IIF approach, the diffusion term is solved exactly while the nonlinear equations resulting from the implicit treatment of reactions are decoupled from the diffusion term to avoid solving large nonlinear systems. As a result, the size of the nonlinear system arising from the implicit treatment is independent of the number of spatial grid points, and the small nonlinear algebraic system can be solved element by element by Picard iteration or Newton iteration.

For a system in high (two or three) spatial dimensions, the dominant computational cost of IIF method arises from the storage and calculation of exponentials of resulting matrices. To deal with this difficulty, two types of approaches were introduced in the context of IIF method. The first one is the Krylov subspace method which approximates the multiplication between the exponential of matrix and vector [15, 19, 20]. The Krylov implicit integration factor method is robust in its implementation with various spatial discretization methods such as FDM, FVM, and DG methods. It also adapts to different mesh generation including triangular and quadrilateral mesh. However, at each time step the Krylov subspace method needs to be carried out at each time step, leading to a significant increase in CPU time.

The other type of approach to avoid storage of the exponentials of matrices is a compact implicit integration factor (cIIF) method [11]. By introducing the compact representation for the matrix approximating the differential operator, the compact IIF methods apply matrix exponential operations sequentially in every spatial direction. As a result, exponential matrices which are calculated and stored have small sizes, as those in the 1D problem. For two or three dimensions, the cIIF method is significantly more efficient in both storage and CPU cost. Recently, based on the idea of cIIF method, an array-representation compact implicit integration (AcIIF) method [16] was proposed for efficient handling of a general linear differential operator that includes cross-derivatives and non-constant diffusion coefficients. Despite the various advantages and tremendous success, the cIIF method has its own shortcomings. A serious drawback of this class of methods is that it is limited to second-order accuracy in space.

In the context of high-order finite differences, compact finite difference methods feature high-order accuracy and smaller stencils [1, 6, 10, 13, 17]. Recently, there has been a renewed interest in the development and application of compact finite difference methods for the numerical solution of the nonlinear Schrodinger equation [2, 18], advection-diffusion equation [7], and generalized RLW equation [9]. It is evident that they are not only accurate and cost effective but also provide easier treatment of boundary conditions. The implicit and AFI methods were usually applied for the stiff ODE system resulting from the cFDM spatial discretization method. However, large global nonlinear systems need to be solved at each time step. Therefore,the number of operations for the nonlinear scheme may be large. Besides that, these time integration methods are limited to second-order accuracy.

In this paper, we combine the cFDM in space discretization and the cIIF method in time discretization to solve reaction–diffusion systems (1). Because there are two "compact" schemes in this method, we will call it double compact (DC) method in this paper for simplification. To build the cFDM, we adopt a compact scheme which equals a combination of nodal derivatives to a combination of nodal values of the function. By introducing a compact representation of the discretized differential operator, the nodal derivatives

are implicitly evaluated by the nodal values of the function. The DC method not only yields fourth-order accuracy in space but also keeps the same stencil as the finite difference scheme. Moreover, the time accuracy, storage, and CPU cost are the same as with the cIIF method.

This paper is organized as follows. In Sect. 2, we explicitly present this double compact method for both two and three dimensions. In Sect. 3, we present some numerical examples to test the accuracy and efficiency of the new method. Conclusions and discussions are given in Sect. 4.

2 Double compact method

2.1 Two dimensions

In this section, we first illustrate the double compact method by applying it to a two-dimensional reaction–diffusion equation

$$\frac{\partial u}{\partial t} = D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + F(u), \quad (x,y) \in \Omega = [a,b] \times [c,d]. \tag{2}$$

The computation domain Ω is discretized into grids described by the set $(x_i, y_j) = (a + ih_x, c + jh_y)$, where $h_x = (b-a)/N_x$, $h_y = (d-c)/N_y$ and $0 \le i \le N_x$, $0 \le j \le N_y$. We will use the following notations for difference operators.

Define the following linear mapping:

$$\begin{aligned}
&\delta_x^2 u_{ij} = \frac{u_{i-1,j} - 2u_{ij} + u_{i+1,j}}{h_x^2}, \qquad \delta_y^2 u_{ij} = \frac{u_{i,j-1} - 2u_{ij} + u_{i,j+1}}{h_y^2}, \\
&\mathcal{L}_x u_{ij} = \left(1 + \frac{h_x^2}{12}\delta_x^2\right) u_{ij} = \frac{u_{i-1,j} + 10u_{ij} + u_{i+1,j}}{12}, \\
&\mathcal{L}_y u_{ij} = \left(1 + \frac{h_y^2}{12}\delta_y^2\right) u_{ij} = \frac{u_{i,j-1} + 10u_{ij} + u_{i,j+1}}{12}.
\end{aligned} \tag{3}$$

Setting $v = \frac{\partial^2 u}{\partial x^2}$ and $w = \frac{\partial^2 u}{\partial y^2}$, we get the discretization for (2) on the mesh as follows:

$$\frac{d}{dt} u_{ij} = D(v_{ij} + w_{ij}) + F(u_{ij}), \quad 1 \le i \le N_x, 1 \le j \le N_y. \tag{4}$$

We rewrite the nodal values u_{ij} as a matrix form instead of a vector form and define

$$\mathbf{U} = (u_i, j)_{N_x \times N_y} = \begin{pmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,N_y} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,N_y} \\ \vdots & \vdots & \vdots & \vdots \\ u_{N_x,1} & u_{N_x,2} & \cdots & u_{N_x,N_y} \end{pmatrix}. \tag{5}$$

Similarly, $\mathbf{V}$, $\mathbf{W}$ are defined as an $N_x \times N_y$ matrix for the nodal derivatives v_{ij} and w_{ij}. The semi-discretized form (4) can be written in terms of matrices

$$\frac{d}{dt}\mathbf{U} = D(\mathbf{V} + \mathbf{W}) + \mathbf{F}(\mathbf{U}). \tag{6}$$

Every element in matrix $\mathbf{F}(\mathbf{U})$ is defined as $F(u_{ij})$, i.e., that $\mathbf{F}(\mathbf{U}) = (F(u_{ij}))_{N_x \times N_y}$.

Next we will build the linear mapping between the derivative matrices $\mathbf{V}$, $\mathbf{W}$ and the solution matrix $\mathbf{U}$. By using a Taylor expansion, we get

$$\begin{aligned}
&\delta_x^2 u_{ij} = v_{ij} + \frac{h_x^2}{12}\delta_x^2 v_{ij} + \mathcal{O}\big(h^4\big) = \mathcal{L}_x v_{ij} + \mathcal{O}\big(h^4\big),\\
&\delta_y^2 u_{ij} = w_{ij} + \frac{h_y^2}{12}\delta_y^2 w_{ij} + \mathcal{O}\big(h^4\big) = \mathcal{L}_y w_{ij} + \mathcal{O}\big(h^4\big),
\end{aligned}$$

where $h = \max\{h_x, h_y\}$. Omitting the small terms $\mathcal{O}(h^4)$, we obtain the approximation of v_{ij} and w_{ij}:

$$v_{ij} = \mathcal{L}_x^{-1}\delta_x^2 u_{ij}, \qquad w_{ij} = \mathcal{L}_y^{-1}\delta_y^2 u_{ij}. \tag{7}$$

We define the matrices $\mathbf{A}_m = \frac{D}{h_m^2}\mathbf{A}_{N_m\times N_m}$ and $\mathbf{B}_m = \mathbf{B}_{N_m\times N_m}$, $m = x, y$, where

$$\begin{aligned}
\mathbf{A}_{N\times N} &= \begin{pmatrix} -2 & 1 & 0 & \cdots & 1 \\ 1 & -2 & 1 & \cdots & 0 \\ 0 & 1 & -2 & 1 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 1 & 0 & \cdots & 1 & -2 \end{pmatrix}_{N\times N},\\
\mathbf{B}_{N\times N} &= \begin{pmatrix} \frac{5}{6} & \frac{1}{12} & 0 & \cdots & \frac{1}{12} \\ \frac{1}{12} & \frac{5}{6} & \frac{1}{12} & \cdots & 0 \\ 0 & \frac{1}{12} & \frac{5}{6} & \frac{1}{12} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{1}{12} & 0 & \cdots & \frac{1}{12} & \frac{5}{6} \end{pmatrix}_{N\times N}.
\end{aligned} \tag{8}$$

Then the linear mapping equation (7) can be rewritten as a matrix form

$$\mathbf{V} = \big(\mathbf{B}_x^{-1}\mathbf{A}_x\big)\mathbf{U}, \qquad \mathbf{W} = \mathbf{U}\big(\mathbf{A}_y\mathbf{B}_y^{-1}\big). \tag{9}$$

Substitution of (9) into (6) yields

$$\frac{d}{dt}\mathbf{U} = \big(\mathbf{B}_x^{-1}\mathbf{A}_x\big)\mathbf{U} + \mathbf{U}\big(\mathbf{A}_y\mathbf{B}_y^{-1}\big) + \mathbf{F}(\mathbf{U}). \tag{10}$$

Assume that the final time is $t = T$, and let the time step $\Delta t = T/N$, $t_n = n\Delta t$, $0 \le n \le N$. To construct the cIIF method for (10), we multiply it by the integration factor $e^{-\mathbf{B}_x^{-1}\mathbf{A}_x t}$ from the left and $e^{-\mathbf{A}_y\mathbf{B}_y^{-1}t}$ from the right and integrate over one time step from t_n to t_{n+1} to obtain

$$\begin{aligned}
\mathbf{U}_{n+1} &= e^{\mathbf{B}_x^{-1}\mathbf{A}_x\Delta t}\mathbf{U}_n e^{\mathbf{A}_y\mathbf{B}_y^{-1}\Delta t}\\
&\quad + e^{\mathbf{B}_x^{-1}\mathbf{A}_x\Delta t}\left(\int_0^{\Delta t} e^{-\mathbf{B}_x^{-1}\mathbf{A}_x\tau}\mathbf{F}\big(\mathbf{U}(t_n+\tau)\big)e^{\mathbf{A}_y\mathbf{B}_y^{-1}\tau}\,d\tau\right)e^{\mathbf{A}_y\mathbf{B}_y^{-1}\Delta t}.
\end{aligned} \tag{11}$$

Then we approximate the integrand in (11) by using an $r-1$th-order Lagrange interpolation polynomial with interpolation points at $t_{n+1}, t_n, \ldots, t_{n-r+2}$, and obtain a double com-

pact scheme at the order of $\mathcal{O}(h^4 + \Delta t^r)$:

$$\mathbf{U}_{n+1} = e^{\mathbf{B}_x^{-1}\mathbf{A}_x\Delta t}\mathbf{U}_n e^{\mathbf{A}_y\mathbf{B}_y^{-1}\Delta t} + \Delta t\left(\alpha_1\mathcal{F}(\mathbf{U}_{n+1}) + \sum_{j=0}^{r-2}\alpha_{-j}e^{(j+1)\mathbf{B}_x^{-1}\mathbf{A}_x\Delta t}\mathcal{F}(\mathbf{U}_{n-j})e^{(j+1)\mathbf{A}_y\mathbf{B}_y^{-1}\Delta t}\right). \tag{12}$$

See [12] for the values of coefficients α_j, $j = 1, 0, \ldots, 2 - r$, for the schemes with different orders. For example, the second-order double compact (DC2) scheme with $\mathcal{O}(h^4 + \Delta t^2)$ is of the following form:

$$\mathbf{U}_{n+1} = e^{\mathbf{B}_x^{-1}\mathbf{A}_x\Delta t}\left(\mathbf{U}_n + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U}_n)\right)e^{\mathbf{A}_y\mathbf{B}_y^{-1}\Delta t} + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U}_{n+1}). \tag{13}$$

To solve the nonlinear equation (13), we use the following Picard iterative method:

$$\mathbf{U}_{n+1,l+1} = e^{\mathbf{B}_x^{-1}\mathbf{A}_x\Delta t}\left(\mathbf{U}_n + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U}_n)\right)e^{\mathbf{A}_y\mathbf{B}_y^{-1}\Delta t} + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U}_{n+1,l}), \quad l = 0, 1, \ldots. \tag{14}$$

The initial value of iteration is chosen as $\mathbf{U}_{n+1,0} = \mathbf{U}_n$. The iteration terminated when $\|\mathbf{U}_{n+1,l+1} - \mathbf{U}_{n+1,l+1}\|_\infty < \epsilon$. We take the iteration threshold in the numerical experiments as $\epsilon = 10^{-13}$.

Remark The novel property of the DC method is that the exact evaluation of the diffusion terms is decoupled from the implicit treatment of the nonlinear terms. As a result, only a local nonlinear system needs to be solved at each spatial grid point. The numerical tests show that the method is advantageous in both CPU time and memory savings.

We also consider the fourth-order case such that the order of accuracy in the spatial direction is consistent with the temporal accuracy. The values of α_j, $j = 1, 0, -1, -2$, for the fourth-order double compact (DC4) scheme are defined as $\alpha_1 = \frac{9}{24}$, $\alpha_0 = \frac{19}{24}$, $\alpha_{-1} = -\frac{5}{24}$, and $\alpha_{-2} = \frac{1}{24}$.

Remark The scheme DC4 is multi-step methods. To start the computations at the first few time steps, we use the Runge–Kutta methods. Specifically, the fourth-order Runge–Kutta method is used for the first U^1 and the second time steps U^2 in DC4. Coupled with initial value U^0, the scheme DC4 evolves with time.

2.2 Stability analysis

We study the linear stability of second-order DC methods and discuss the computational costs of the methods in this subsection. And first we give some definitions and lemmas for the stability analysis.

A matrix in the form of

$$\mathbf{C}_{N\times N} = \begin{pmatrix} a_0 & a_0 & \cdots & a_{N-1} \\ a_{N-1} & a_0 & \cdots & a_{N-2} \\ \cdots & \cdots & \cdots & \cdots \\ a_1 & a_2 & \cdots & a_0 \end{pmatrix} \tag{15}$$

is called a circulant matrix [5]. Matrix **C** is determined by the entries in the first row $(a_0, a_0, \ldots, a_{N-1})$. It is clear that matrices **A** and **B** are circulant matrices.

The circulant matrix has some useful properties as follows [5]:

- If a circulant matrix **C** is invertible, then its inverse matrix $\mathbf{C}^{-1}$ is circulant.
- Circulant matrices satisfy the operator commuting since, for any two given circulant matrices **C** and **D**, the product **CD** is circulant, and **CD** = **DC**.
- For a real circulant matrix **C** in (15), all eigenvalues of **C** are given by $\lambda = a_0 + a_1\omega_k + a_2\omega_k^2 + \cdots + a_{N-1}\omega_k^{N-1}$ with $\omega_k = \exp(\frac{i2\pi k}{N})$, $k = 0, 1, 2, \ldots, N-1$.

The proceeding properties give the eigenvalues of the $N \times N$ order circulant matrices **A** and **B** in the form of

$$\lambda_k^{\mathbf{A}} = -2 + 2\cos\left(\frac{2\pi k}{N}\right), \qquad \lambda_k^{\mathbf{B}} = \frac{5}{6} + \frac{1}{6}\cos\left(\frac{2\pi k}{N}\right), \quad k = 0, 1, \ldots, N-1. \tag{16}$$

The eigenvalues indicate that matrix $-\mathbf{A}$ is a positive semi-definite, symmetric, and circulant matrix, and matrix $\mathbf{B}^{-1}$ is a positive definite, symmetric, and circulant matrix. With the first and the second property of a circulant matrix, we can get $-\mathbf{B}^{-1}\mathbf{A} = -\mathbf{A}\mathbf{B}^{-1}$. This commutativity indicates that $-\mathbf{B}^{-1}\mathbf{A}$ is positive semi-definite and its eigenvalues are non-negative. Based on linear stability analyses in [11], we claim that the second-order DC method, Eq. (13), is asymptotically stable for the case of $\mathcal{F}(u) = du$ and $\mathcal{L}_x^{-1}\delta_x^2 u = -cu$, where $d < 0$ and $c > 0$ correspond to stable reactions and elliptic operators. For more details on the stability analysis, the reader is referred to the analysis in a unified framework [11].

In comparison with the non-compact FDM spatial discretization coupled with the compact IIF method, extra work for the DC method is the computation of the inverse of matrix $\mathbf{B}^{-1}$. Since matrix **B** has small order of magnitude only with $N \times N$, the computation of inverse matrix is not CPU-intensive. In our numerical tests, the size of matrix B is 128 or 256. The computation for the inverse of this matrix could be easily implemented by inv(B) in Matlab. In addition, the exponential matrices such as $e^{\mathbf{B}^{-1}\mathbf{A}_x\Delta t}$ are pre-computed and stored for later use at every time step. Therefore the new method is advantageous in accuracy without increasing both CPU time and memory savings.

2.3 Three dimensions

In this section we extend the double compact representation of the Laplacian operator to three-dimensional systems. In this section, we present a derivation for a three-dimensional reaction–diffusion equation in a cube with periodic boundary conditions:

$$\frac{\partial u}{\partial t} = D\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + F(u), \quad (x, y) \in \Omega = [a_l, a_u] \times [b_l, b_u] \times [c_l, c_u]. \tag{17}$$

Let N_x, N_y, N_z be the number of spatial grid points in each spatial direction and h_x, h_y, h_z be the grid size, respectively, and $u_{i,j,k}$ represents the approximate solution at the grid point (ih_x, jh_y, kh_z).

Setting $v = \frac{\partial^2 u}{\partial x^2}$, $w = \frac{\partial^2 u}{\partial y^2}$, and $\phi = \frac{\partial^2 u}{\partial z^2}$, we get the discretization of (17) on the grid point as follows:

$$\frac{d}{dt}u_{ijk} = D(v_{ijk} + w_{ijk} + \phi_{ijk}) + F(u_{ijk}), \quad 1 \le i \le N_x, 1 \le j \le N_y, 1 \le k \le N_z. \tag{18}$$

Define the following linear mapping in three dimensions:

$$\begin{aligned} &\delta_x^2 u_{i,j,k} = D\frac{u_{i-1,j,k} - 2u_{i,j,k} + u_{i+1,j,k}}{h_x^2}, \\ &\mathcal{L}_x v_{i,j,k} = \left(1 + \frac{h_x^2}{12}\delta_x^2\right)u_{i,j,k} = \frac{u_{i-1,j,k} + 10u_{i,j,k} + u_{i+1,j,k}}{12}. \end{aligned} \tag{19}$$

The linear mappings $\delta_y^2 u_{i,j,k}$, $\delta_z^2 u_{i,j,k}$, and $\mathcal{L}_y v_{i,j,k}$, $\mathcal{L}_z v_{i,j,k}$ are similarly defined. Based on the approximation (7), we can get $v_{i,j,k} = \mathcal{L}_x^{-1}\delta_x^2 u_{i,j,k}$, $w_{i,j,k} = \mathcal{L}_y^{-1}\delta_y^2 u_{i,j,k}$, $\phi_{i,j,k} = \mathcal{L}_z^{-1}\delta_x^2 u_{i,j,k}$. Define the three-dimensional array: $\mathbf{U} = (u_{i,j,k}|i = 1,\dots,N_x, j = 1,\dots,N_y, k = 1,\dots,N_z)$. The fourth-order compact finite difference scheme for Eq. (18) takes the form

$$\frac{d}{dt}\mathbf{U} = \mathcal{L}_x^{-1}\delta_x^2\mathbf{U} + \mathcal{L}_y^{-1}\delta_y^2\mathbf{U} + \mathcal{L}_z^{-1}\delta_z^2\mathbf{U} + \mathcal{F}(\mathbf{U}). \tag{20}$$

When the second-order IIF is applied to the reaction–diffusion equations of the system of Eq. (20), we obtain

$$\mathbf{U}_{n+1} = e^{(\mathcal{L}_x^{-1}\delta_x^2 + \mathcal{L}_y^{-1}\delta_y^2 + \mathcal{L}_z^{-1}\delta_z^2)\Delta t}\left(\mathbf{U}_n + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U_n})\right) + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U_{n+1}}). \tag{21}$$

To avoid computing the exponential of a huge matrix, we adopt the array-representation implicit integration factor (AcIIF) method which decomposes the matrix into small matrices based on an array representation. See [16] for more details. The three-dimensional array $\mathbf{U}$ can be treated as a collection of all such one-dimensional vectors on a two-dimensional array

$$\mathbf{U} = \bigotimes_{\substack{j=1,\dots,N_y \\ k=1,\dots,N_z}} \mathbf{U}(:,j,k), \tag{22}$$

where $\mathbf{U}(:,j,k)$ is a vector by fixing the last two indices j, k, $\mathbf{U}(:,j,k) = (u_{1,j,k}, u_{2,j,k},\dots, u_{N_x,j,k})^T$. With the definition of matrices $\mathbf{A}_x$ and $\mathbf{B}_x$, the exponential of linear mapping $\mathcal{L}_x^{-1}\delta_x^2$ in the array representation can be written as a matrix form:

$$e^{\mathcal{L}_x^{-1}\delta_x^2}\mathbf{U} = \bigotimes_{\substack{j=1,\dots,N_y \\ k=1,\dots,N_z}} e^{\mathbf{B}_x^{-1}\mathbf{A}_x}\mathbf{U}(:,j,k). \tag{23}$$

The exponentials of linear mappings $\mathcal{L}_y^{-1}\delta_y^2$ and $\mathcal{L}_z^{-1}\delta_z^2$ have similar array representations.

Because the linear mappings $\mathcal{L}_x^{-1}\delta_x^2$, $\mathcal{L}_y^{-1}\delta_y^2$, and $\mathcal{L}_z^{-1}\delta_z^2$ satisfy the commutativity based on the property of a circulant matrix, we get

$$e^{\mathcal{L}_x^{-1}\delta_x^2 + \mathcal{L}_y^{-1}\delta_y^2 + \mathcal{L}_z^{-1}\delta_z^2} = e^{\mathcal{L}_x^{-1}\delta_x^2}e^{\mathcal{L}_y^{-1}\delta_y^2}e^{\mathcal{L}_z^{-1}\delta_z^2}. \tag{24}$$

Direct application of Eqs. (23) and (24) to Eq. (21) results in the following double compact method with order $\mathcal{O}(h^4 + \Delta t^2)$:

$$\mathbf{U}_{n+1} = \bigotimes_{\substack{j=1,\dots,N_y \\ k=1,\dots,N_z}} e^{\mathbf{B}_x^{-1}\mathbf{A}_x} \left(\bigotimes_{\substack{i=1,\dots,N_x \\ k=1,\dots,N_z}} e^{\mathbf{B}_y^{-1}\mathbf{A}_y} \left(\bigotimes_{\substack{i=1,\dots,N_x \\ j=1,\dots,N_y}} e^{\mathbf{B}_z^{-1}\mathbf{A}_z} \boldsymbol{\Psi}(i,j,:) \right)(i,:,k) \right)(:,j,k) + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U_{n+1}}), \tag{25}$$

where $\boldsymbol{\Psi} = \mathbf{U} + \frac{\Delta t}{2}\mathcal{F}(\mathbf{U_n})$.

3 Numerical experiments

In this section, we demonstrate the performance of the proposed double compact scheme on a number of test problems. Firstly, we test our scheme for a linear reaction–diffusion equation with exact solution. In this test, we investigate the accuracy and efficiency of our new scheme by comparison with other methods such as the second-order Runge–Kutta method and the original cIIF method. Then we apply the scheme to the chloride-iodide-malonic acid (CIMA) model which was derived by Lengyel and Epstein [8]. It can be found that different choice of dimensionless parameters will lead to a different pattern [19].

Compared with the cIIF method, extra computation is the inverse of matrices $\mathbf{B}_x$ and $\mathbf{B}_y$. Because these matrices depend only on the spatial grid size in every spatial direction, these matrices have small sizes as those in 1D problem and the inverse of matrices can be easily computed. As the cIIF method, for a given spatial and temporal numerical resolution, the exponential matrices are pre-computed and stored for later use at every time step.

3.1 The accuracy test

Example 1 We consider the following linear reaction–diffusion equation on a rectangle $\Omega = [0, 2\pi]^2$:

$$\frac{\partial u}{\partial t} = 0.2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + 0.1u \tag{26}$$

with periodic boundary conditions. The exact solution of this equation is $u = e^{-0.1t}(\cos(x) + \cos(y))$. The initial condition is determined by the exact solution. The final computation time is $t = 1$. The time step is proportional to the spatial grid size, here we choose $\Delta t = 1/N_x$. The L^∞ error is measured by difference between the numerical solution and the exact solution. For the convenience of comparison, we solve this problem by the second-order double compact (DC2) method, the second-order cIIF (cIIF2) method, and the second-order Runge–Kutta (RK2) method. The error, order of accuracy, and CPU time for three methods are listed in Table 1. As seen in the table, DC2 is more accurate than cIIF2 without adding any computational cost. On the other hand, RK2 demands a much smaller time step because of stability constraint.

Because the time discretization only has the second order, we cannot get the fourth-order convergence overall. Now we consider a fourth-order double compact (DC4) scheme in an attempt to balance the spatial and temporal accuracy of the overall scheme. The DC4 scheme is a multi-step method. We start the computations at the first time step $\mathbf{U}_1$ and the second time step $\mathbf{U}_2$. In our numerical simulation we use the DC2 scheme with

Table 1 Error, order of accuracy, and CPU time with DC2, cIIF2, and RK2 schemes for Example 1

$N_x \times N_y$	DC2, $\Delta t = 1/N_x = h_x/2\pi$			cIIF2, $\Delta t = 1/N_x = h_x/2\pi$			RK2, $\Delta t = h_x^2$		
	L^∞ error	Order	CPU (s)	L^∞ error	Order	CPU (s)	L^∞ error	Order	CPU (s)
32×32	2.39×10^{-6}	–	0.00	1.16×10^{-3}	–	0.00	1.16×10^{-3}	–	0.00
64×64	1.77×10^{-7}	3.76	0.03	2.91×10^{-4}	2.00	0.03	2.91×10^{-4}	2.00	0.07
128×128	1.80×10^{-8}	3.30	0.37	7.27×10^{-5}	2.00	0.37	7.27×10^{-5}	2.00	2.20
256×256	2.85×10^{-9}	2.66	5.71	1.82×10^{-5}	2.00	5.68	1.82×10^{-5}	2.00	70.5

Table 2 Error, order of accuracy, and CPU time with DC4 scheme for Example 1

$N_x \times N_y$	Δt	L^∞ error	Order	CPU (s)
32×32	1/32	2.2449×10^{-6}	–	0.0058
64×64	1/64	1.4015×10^{-7}	4.0016	0.0658
128×128	1/128	8.7572×10^{-9}	4.0004	1.0121
256×256	1/256	5.4901×10^{-10}	3.9956	16.578

Table 3 Error, order of accuracy, and CPU time with DC2, cIIF2, and RK2 schemes for Example 2

$N_x \times N_y \times N_z$	DC2, $\Delta t = 1/N_x = h_x/2\pi$			cIIF2, $\Delta t = 1/N_x = h_x/2\pi$			RK2, $\Delta t = h_x^2/3$		
	L^∞ error	Order	CPU (s)	L^∞ error	Order	CPU (s)	L^∞ error	Order	CPU (s)
$16 \times 16 \times 16$	5.50×10^{-5}	–	0.05	6.95×10^{-3}	–	0.04	6.95×10^{-3}	–	0.09
$32 \times 32 \times 32$	3.59×10^{-6}	3.94	0.43	1.74×10^{-3}	2.00	0.43	1.74×10^{-3}	2.00	2.06
$64 \times 64 \times 64$	2.65×10^{-7}	3.76	5.93	4.36×10^{-4}	2.00	5.95	4.36×10^{-4}	2.00	61.9
$128 \times 128 \times 128$	2.69×10^{-8}	3.30	127	1.09×10^{-4}	2.00	124	1.09×10^{-4}	2.00	2448

a time step $\Delta t = 1/N_x^2$ to $\mathbf{U}_1$ and $\mathbf{U}_2$. Then we go ahead to simulate the problem using the DC4 scheme with time step $\Delta t = 1/N_x$. The error, order of accuracy, and CPU time for the DC4 scheme are listed in Table 2. We can see that the solution by the DC4 scheme is fourth-order accurate in the temporal and spatial dimensions with time step $\Delta t = 1/N_x$. The DC4 scheme only triples the CPU time over the DC2 scheme in the meantime.

Example 2 Then we consider a similar system in three dimensions on $\Omega = [0, 2\pi]^2$:

$$\frac{\partial u}{\partial t} = 0.2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right) + 0.1u \tag{27}$$

with periodic boundary conditions. The exact solution of this equation is

$$u = e^{-0.1t}\big(\cos(x) + \cos(y) + \cos(z)\big).$$

The final computation time is $t = 1$ at which the L^∞ error is measured. The time step is chosen as $\Delta t = 1/N_x$. Similar to the two-dimensional case, we list the error, order of accuracy, and CPU time for DC2, cIIF2, and RK2 methods in Table 3. As shown in Table 3, the DC2 scheme achieves higher-order accuracy while requires the same CPU time as cIIF2. The RK2 method requires a much smaller time step and becomes more expensive. Especially for large grid number $N = 128$, the computation time is too long to be acceptable. The solution by the DC4 scheme for the three-dimensional case with time step $\Delta t = 1/N_x$ is shown in Table 4. As in the two-dimensional case, we can see that the solution by the DC4 scheme is fourth-order accurate in the temporal and spatial dimensions.

Table 4 Error, order of accuracy, and CPU time with DC4 scheme for Example 2

$N_x \times N_y$	Δt	L^∞ error	Order	CPU (s)
$16 \times 16 \times 16$	1/16	5.4123×10^{-5}	–	0.1725
$32 \times 32 \times 32$	1/32	3.3674×10^{-6}	4.0065	1.9944
$64 \times 64 \times 64$	1/64	2.1022×10^{-7}	4.0017	29.632
$128 \times 128 \times 128$	1/128	1.3136×10^{-8}	4.0003	640.60

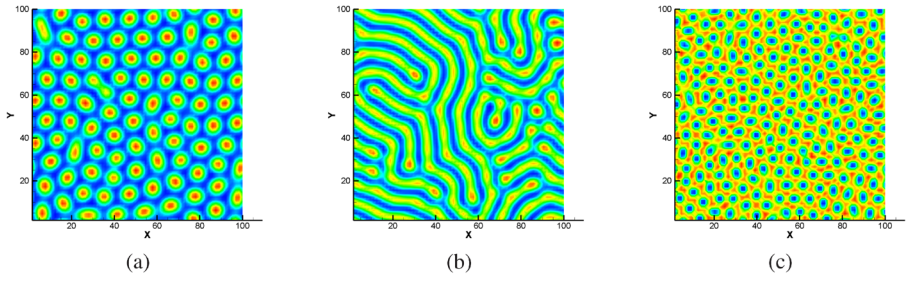

Figure 1 H_0 hexagon, stripe, and H_π hexagon pattern of CIMA model at $t = 10^4$ for 2D

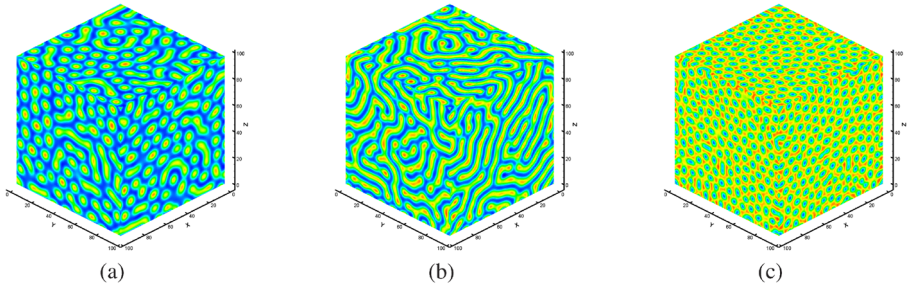

Figure 2 H_0 hexagon, stripe, and H_π hexagon pattern of CIMA model at $t = 10^4$ for 3D

3.2 CIMA model

Example 3 Lengyel and Epstein proposed a two-variable kinetic mechanism for CIMA reaction. In this skeleton version, iodide and chlorite play respectively the roles of the activator and the inhibitor:

$$\begin{aligned} &\frac{\partial u}{\partial t} = D_u \nabla^2 u + \frac{1}{\sigma}\left(a - u - 4\frac{uv}{1+u^2}\right), \\ &\frac{\partial v}{\partial t} = D_v \nabla^2 v + b\left(u - \frac{uv}{1+u^2}\right), \end{aligned} \tag{28}$$

where $D_u = \frac{1}{\sigma}$, $D_v = d$, $d = 1.07$, and $\sigma = 50$. We will solve the CIMA model on two-dimensional case and three-dimensional case. The domains on 2D and 3D are chosen as $\Omega = [0, 100]^2$ and $[0, 100]^3$, respectively. In our computation, we choose the mesh as 64×64 and $64 \times 64 \times 64$. Random initial concentration distributions of both species are used. In the simulation the initial condition is taken as $u = 10^{-1}\,\mathrm{rand}(\cdots)$, $v = 10^{-1}\,\mathrm{rand}(\cdots)$, where $\mathrm{rand}(\cdots)$ is a random function in Fortran [19].

The Turing pattern needs a long computation time to appear. Here we set the final computation time as $t = 10{,}000$. The different patterns will be obtained by selecting three sets

of values for parameters a, b. The first set ($a = 8.8$, $b = 0.09$) leads to an H_0 hexagon pattern as shown in Fig. 1(a). The second set ($a = 10$, $b = 0.16$) gives rise to a stripe pattern (see Fig. 1(b)). The third set ($a = 12$, $b = 0.39$) generates an H_π hexagon pattern plotted in Fig. 1(c). Simulations for these three sets of parameters have also been presented in the 3D case. We observe similar patterns by selecting the corresponding parameters, see Fig. 2.

4 Concluding remarks

In this paper, we combined the compact FDM in space and the compact IIF method in time to propose a high-order accurate method for solving reaction–diffusion equation. The global accuracy order of this method is $\mathcal{O}(h^4 + \Delta t^r)$ ($r = 2, 3, 4, \ldots$) and it allows a considerable saving in the computation time as the cIIF method. The numerical experiments are conducted to show its superiority over the classical RK method and cIIF method. In the future work we plan to apply the DC scheme for solving the variable coefficients reaction–diffusion problem.

Acknowledgements
The authors would like to thank the referees for their valuable comments and suggestions.

Funding
This work was supported by the National Natural Science Foundation of China (No. 61573008, 61703290), Science Technology on Reliability Environmental Engineering Laboratory (No. 6142A0502020717) and SDUST Research Fund (No. 2014TDJH102).

Availability of data and materials
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
The authors declare that the study was realized in collaboration with the same responsibility. All authors read and approved the final manuscript.

Author details
[1]School of Mathematics and Systematic Sciences, Shenyang Normal University, Shenyang, P.R. China. [2]College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao, P.R. China. [3]Department of Foreign Language, Shenyang Normal University, Shenyang, P.R. China.

Received: 6 April 2018 Accepted: 30 July 2018 Published online: 08 August 2018

References

1. Dlamini, P.G., Motsa, S.S., Khumalo, M.: On the comparison between compact finite difference and pseudospectral approaches for solving similarity boundary layer problems. Math. Probl. Eng. **2013**, Article ID 746489 (2013)
2. Gao, Z., Xie, S.: Fourth-order alternating direction implicit compact finite difference schemes for two-dimensional Schrödinger equations. Appl. Numer. Math. **61**, 593–614 (2011)
3. Gierer, A., Meinhardt, H.: A theory of biological pattern formation. Kybernetik **12**, 30–39 (1972)
4. Gray, P., Scott, S.K.: Autocatalytic reactions in the isothermal, continuous stirred tank reactor: isolas and other forms of multistability. Chem. Eng. Sci. **38**(1), 29–43 (1983)
5. Gray, R.M.: Toeplitz and circulant matrices. ISL, Tech. Rep., Stanford Univ., Stanford, CA. http://ee-www.stanford.edu/gray/toeplitz.html (2002)
6. Gu, Y., Liao, W., Zhu, J.: An efficient high-order algorithm for solving systems of 3-D reaction–diffusion equations. J. Comput. Appl. Math. **155**(1), 1–17 (2003)
7. Gurarslan, G., Karahan, H., Alkaya, D., Sari, M., Yasar, M.: Numerical solution of advection–diffusion equation using a sixth-order compact finite difference method. Math. Probl. Eng. **2013**, Article ID 672936 (2013)
8. Lengyel, I., Epstein, I.R.: Modeling of Turing structures in the chlorite-iodide-malonic acid-starch reaction system. Science **251**, 650–652 (1991)
9. Li, S., Wang, J., Luo, Y.: A fourth-order conservative compact finite difference scheme for the generalized RLW equation. Math. Probl. Eng. **2015**, Article ID 960602 (2015)

10. Liao, W.: A high-order ADI finite difference scheme for a 3D reaction–diffusion equation with Neumann boundary condition. Numer. Methods Partial Differ. Equ. **29**(3), 778–798 (2013)
11. Nie, Q., Wan, F., Zhang, Y.-T., Liu, X.-F.: Compact integration factor methods in high spatial dimensions. J. Comput. Phys. **227**, 5238–5255 (2008)
12. Nie, Q., Zhang, Y.-T., Zhao, R.: Efficient semi-implicit schemes for stiff systems. J. Comput. Phys. **214**, 521–537 (2006)
13. Sambit, D., Liao, W., Gupta, A.: An efficient fourth order low dispersive finite difference scheme for a 2-D acoustic wave equation. J. Comput. Appl. Math. **258**, 151–167 (2014)
14. Schnakenberg, J.: Simple chemical reaction systems with limit cycle behavior. J. Theor. Biol. **81**, 389–400 (1979)
15. Sidje, R.B.: Expokit: software package for computing matrix exponentials. ACM Trans. Math. Softw. **24**, 130–156 (1998)
16. Wang, D., Zhang, L., Nie, Q.: Array-representation integration factor method for high-dimensional systems. J. Comput. Phys. **258**, 585–600 (2014)
17. Wang, Y., Zhang, H.: Higher-order compact finite difference method for systems of reaction–diffusion equations. J. Comput. Appl. Math. **233**(2), 502–518 (2009)
18. Zhang, R., Liu, J., Zhao, G.: An efficient compact finite difference method for the solution of the Gross–Pitaevskii equation. Adv. Condens. Matter Phys. **2015**, Article ID 127580 (2015)
19. Zhang, R., Yu, X., Zhu, J., Loula, A.F.D.: Direct discontinuous Galerkin method for nonlinear reaction–diffusion systems in pattern formation. Appl. Math. Model. **38**, 1612–1621 (2014)
20. Zhang, R., Yu, X., Zhu, J., Loula, A.F.D., Cui, X.: Weighted interior penalty method with semi-implicit integration factor method for non-equilibrium radiation diffusion equation. Commun. Comput. Phys. **14**, 1287–1303 (2013)